B 7

```
NNN         NNN     IIIIIIIII        CCCCCCCCCCCC NNN         NNN  FFFFFFFFFFFFFFF
NNN         NNN     IIIIIIIII        CCCCCCCCCCCC NNN         NNN  FFFFFFFFFFFFFFF
NNN         NNN     IIIIIIIII        CCCCCCCCCCCC NNN         NNN  FFFFFFFFFFFFFFF
NNN         NNN        III        CCC             NNN         NNN  FFF
NNN         NNN        III        CCC             NNN         NNN  FFF
NNN         NNN        III        CCC             NNN         NNN  FFF
NNNNNN      NNN        III        CCC             NNNNNN      NNN  FFF
NNNNNN      NNN        III        CCC             NNNNNN      NNN  FFF
NNNNNN      NNN        III        CCC             NNNNNN      NNN  FFF
NNN   NNN   NNN        III        CCC             NNN   NNN   NNN  FFFFFFFFFFFF
NNN   NNN   NNN        III        CCC             NNN   NNN   NNN  FFFFFFFFFFFF
NNN   NNN   NNN        III        CCC             NNN   NNN   NNN  FFFFFFFFFFFF
NNN      NNNNNN        III        CCC             NNN      NNNNNN  FFF
NNN      NNNNNN        III        CCC             NNN      NNNNNN  FFF
NNN      NNNNNN        III        CCC             NNN      NNNNNN  FFF
NNN         NNN        III        CCC             NNN         NNN  FFF
NNN         NNN        III        CCC             NNN         NNN  FFF
NNN         NNN        III        CCC             NNN         NNN  FFF
NNN         NNN     IIIIIIIII        CCCCCCCCCCCC NNN         NNN  FFF
NNN         NNN     IIIIIIIII        CCCCCCCCCCCC NNN         NNN  FFF
NNN         NNN     IIIIIIIII        CCCCCCCCCCCC NNN         NNN  FFF
```

**F

```
**FILE**ID**CNFSEND




  CCCCCCCC  NN      NN  FFFFFFFFFF    SSSSSSSS  EEEEEEEEEE  NN      NN  DDDDDDDD
  CCCCCCCC  NN      NN  FFFFFFFFFF    SSSSSSSS  EEEEEEEEEE  NN      NN  DDDDDDDD
CC          NN      NN  FF          SS          EE          NN      NN  DD      DD
CC          NN      NN  FF          SS          EE          NN      NN  DD      DD
CC          NNNN    NN  FF          SS          EE          NNNN    NN  DD      DD
CC          NNNN    NN  FF          SS          EE          NNNN    NN  DD      DD
CC          NN  NN  NN  FFFFFFFF      SSSSSS    EEEEEEEE    NN  NN  NN  DD      DD
CC          NN  NN  NN  FFFFFFFF      SSSSSS    EEEEEEEE    NN  NN  NN  DD      DD
CC          NN    NNNN  FF                  SS  EE          NN    NNNN  DD      DD
CC          NN    NNNN  FF                  SS  EE          NN    NNNN  DD      DD
CC          NN      NN  FF                  SS  EE          NN      NN  DD      DD      ....
CC          NN      NN  FF                  SS  EE          NN      NN  DD      DD      ....
  CCCCCCCC  NN      NN  FF          SSSSSSSS    EEEEEEEEEE  NN      NN  DDDDDDDD        ....
  CCCCCCCC  NN      NN  FF          SSSSSSSS    EEEEEEEEEE  NN      NN  DDDDDDDD        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
%TITLE  'DECnet Ethernet Configurator Module'
MODULE CNFSEND          (
                        LANGUAGE (BLISS32),
                        IDENT = 'V04-000'
                        ) =
BEGIN

!
!****************************************************************************
!*                                                                          *
!*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
!*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
!*  ALL RIGHTS RESERVED.                                                    *
!*                                                                          *
!*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
!*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
!*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
!*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
!*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
!*  TRANSFERRED.                                                            *
!*                                                                          *
!*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
!*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
!*  CORPORATION.                                                            *
!*                                                                          *
!*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
!*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
!*                                                                          *
!*                                                                          *
!****************************************************************************
!

!++
! FACILITY:     DECnet Configurator Module (NICONFIG)
!
! ABSTRACT:
!
!       This module contains the routines to buffer and send NICE
!       response messages to the processes which interrupt with requests.
!
! ENVIRONMENT:  VAX/VMS Operating System
!
! AUTHOR:       Bob Grosso,     CREATION DATE:  18-Jan-1982
!
! MODIFIED BY:
!
!--
```

```
%SBTTL  'Definitions'

!
! INCLUDE FILES:
!

LIBRARY 'SYS$LIBRARY:STARLET';                  ! VMS common definitions

REQUIRE 'LIB$:CNFDEF.R32';

REQUIRE 'SRC$:CNFPREFIX.REQ';


!
! BUILTIN functions
!

BUILTIN
    INSQUE,                                     ! INSQUE instruction
    REMQUE;                                     ! REMQUE instruction

!
! TABLE OF CONTENTS:
!

FORWARD ROUTINE

        CNF$BUFR_NICE_MSG,                      ! Buffer NICE messages into IRB
        CNF$SEND_NICE_MSG;                      ! Send the NICE message stored in IRB

!
! EXTERNAL REFERENCES:
!

EXTERNAL ROUTINE

!     Module CNFMAIN

        CNF$TRACE,                              ! Log messages to log file
        CNF$LOG_DATA,                           ! Log formatted data to log file
        CNF$GET_ZVM,                            ! Get zeroed virtual memory
        CNF$FREE_VM,                            ! Free virtual memory

!     Module CNFINTRPT

        CNF$CLOSE_REQUEST_LINK,                 ! After an unsuccessful IO shut down the link and deallocate control
        CNF$SOLICIT_REQUEST,

!     Module CNFWORKQ

        WKQ$ADD_WORK_ITEM;                      ! Add work to work queue


EXTERNAL LITERAL

        CNF$_LINK,                              ! Error on logical link

```

```
1           CNF$C_ASYNCH_EFN;
1
1 EXTERNAL
1
1           CNF$GL_LOGMASK : BITVECTOR [32];
1
```

```
1 %SBTTL  'CNF$BUFR_NICE_MSG  Buffer the response message'
1 GLOBAL ROUTINE CNF$BUFR_NICE_MSG (IRB, MSG, DEALLOCATE_LEN) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !  Place the NICE message onto a linked list of messages stored
1 !  in the IRB for later transmission to the connectee.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       irb             Interrupt Request Block, contains context for
1 !                       I/O with connectee.
1 !
1 !       msg             address of buffer containing NICE message to be
1 !                       stored in the IRB.
1 !
1 !       deallocate_len  Length of message to be deallocated after transmission.
1 !                       Some messages are stored in buffers allocated in VM
1 !                       and must be deallocated after transmission.  Others
1 !                       reside on the stack or in OWN storage and shouldn't
1 !                       be deallocated.
1 !
1 ! IMPLICIT INPUTS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !       Success
1 !
1 !--
1
2     BEGIN
2     MAP
2         IRB : REF BBLOCK,
2         MSG : REF BBLOCK;
2
2     LOCAL
2         BNR : REF BBLOCK,
2         STATUS;
2
2
2     CNF$TRACE (DBG$C_TRACE, $DESCRIPTOR('TRACE'),
2         $DESCRIPTOR ('cnf$bufr_nice_msg'));
2
2
2     EXECUTE (CNF$GET_ZVM (%REF (BNR$C_LENGTH), BNR) );           ! Get space to store header and message
2     BNR [BNR$L_ADDRESS] = .MSG [DSC$A_POINTER];                  ! Record message buffer pointer
2     BNR [BNR$W_LENGTH] = .MSG [DSC$W_LENGTH];                    ! Record message length
2
2     BNR [BNR$W_FREE_LEN] = .DEALLOCATE_LEN;
2     INSQUE (.BNR, .IRB [IRB$L_BNR_BLINK]);                       ! Queue message onto IRB
2
2     RETURN TRUE;
```

```
;  171             0355  1      END;                                                        ! Routine cnf$bufr_nice_msg

                                                                              .TITLE   CNFSEND DECnet Ethernet Configurator Module
                                                                              .IDENT   \V04-000\

                                                                              .PSECT   $PLIT$,NOWRT,NOEXE,2

                                       45  43  41  52  54  00000 P.AAB:       .ASCII   \TRACE\                                   ;
                                                           00005              .BLKB    3
                                             00000005      00008 P.AAA:       .LONG    5                                         ;
                                             00000000'     0000C              .ADDRESS P.AAB                                     ;
6D  5F  65  63  69  6E  5F  72  66  75  62  24  66  6E  63  00010 P.AAD:      .ASCII   \cnf$bufr_nice_msg\                       ;
                                                   67  73  0001F                                                                 ;
                                                           00021              .BLKB    3
                                             00000011      00024 P.AAC:       .LONG    17                                        ;
                                             00000000'     00028              .ADDRESS P.AAD                                     ;

                                                                              .EXTRN   CNF$TRACE, CNF$LOG_DATA
                                                                              .EXTRN   CNF$GET_ZVM, CNF$FREE_VM
                                                                              .EXTRN   CNF$CLOSE_REQUEST_LINK
                                                                              .EXTRN   CNF$SOLICIT_REQUEST
                                                                              .EXTRN   WKQ$ADD_WORK_ITEM
                                                                              .EXTRN   CNF$_LINK, CNF$C_ASYNCH_EFN
                                                                              .EXTRN   CNF$GL_LOGMASK

                                                                              .PSECT   $CODE$,NOWRT,2

                                                     0000  00000              .ENTRY   CNF$BUFR_NICE_MSG, Save nothing           ; 0299
                            5E                 08  C2  00002              SUBL2    #8, SP                                    ;
                                        0000'  CF  9F  00005              PUSHAB   P.AAC                                     ; 0344
                                        0000'  CF  9F  00009              PUSHAB   P.AAA                                     ; 0343
                                               01  DD  0000D              PUSHL    #1                                        ;
                     0000G  CF                 03  FB  0000F              CALLS    #3, CNF$TRACE                             ;
                                           04  AE  9F  00014              PUSHAB   BNR                                       ; 0347
                        04  AE                 10  D0  00017              MOVL     #16, 4(SP)                                ;
                                           04  AE  9F  0001B              PUSHAB   4(SP)                                     ;
                     0000G  CF                 02  FB  0001E              CALLS    #2, CNF$GET_ZVM                           ;
                            21                 50  E9  00023              BLBC     STATUS, 1$                                ;
                            51             04  AE  D0  00026              MOVL     BNR, R1                                   ; 0348
                            50             08  AC  D0  0002A              MOVL     MSG, R0                                   ;
                        0C  A1             04  A0  D0  0002E              MOVL     4(R0), 12(R1)                             ;
                        08  A1                 60  B0  00033              MOVW     (R0), 8(R1)                               ; 0349
                        0A  A1             0C  AC  B0  00037              MOVW     DEALLOCATE_LEN, 10(R1)                    ; 0351
                            50             04  AC  D0  0003C              MOVL     IRB, R0                                   ; 0352
                        18  B0                 61  0E  00040              INSQUE   (R1), @24(R0)                             ;
                            50                 01  D0  00044              MOVL     #1, R0                                    ; 0354
                                                   04  00047 1$:          RET                                                ; 0355

; Routine Size:  72 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 %SBTTL  'CNF$BUFR_ERR_MSG  Buffer the error response message'
1 GLOBAL ROUTINE CNF$BUFR_ERR_MSG
1         (IRB, ERR_CODE, ERR_DETAIL, ERR_TXT_COD, ERR_TXT_DSC) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !  Build the error response message and buffer it for later return to
1 !  the connectee.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       irb             Interrupt Request Block, contains context for
1 !                       I/O with connectee.
1 !
1 !       err_code        The error code is returned in the first byte
1 !                       of the NICE response message.
1 !
1 !       err_detail      The error detail is returned in second and third bytes
1 !                       of the NICE response message.
1 !
1 !       err_txt_cod     An optional error status, for which the error text
1 !                       will be obtained and buffered
1 !
1 !       err_txt_dsc     An optional error text which will be buffered
1 !
1 ! IMPLICIT INPUTS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !       Always return success
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Error message is built and buffered and stored in the IRB
1 !
1 !--
1
2     BEGIN
2     BUILTIN
2         NULLPARAMETER;                  ! To check for optional parameters
2
2     LITERAL
2         DECODED_TXT_BUFLEN = 256;       ! Maximum size of text string for decoded error messages
2
2     MAP
2         ERR_TXT_DSC : REF BBLOCK;
2
2     LOCAL
2         ERR_TXTLEN,                 ! Either the length of the text decoded from the ERR_TXT_COD
2                                     !  or the length of optional text in ERR_TXT_DSC
2         MSG :                       ! Descriptor of message being built
2           BBLOCK [DSC$C_S_BLN],
  2              STATUS,
  2              DECODED_TXT_LEN,          ! Length of message text decoded from ERR_TXT_COD
  2              DECODED_TXT_BUFDSC :      ! Descriptor of message text decoded from ERR_TXT_COD
  2                BBLOCK [DSC$C_S_BLN],
  2              DECODED_TXT_BUF :         ! Buffer for message text decoded from ERR_TXT_COD
  2                BBLOCK [DECODED_TXT_BUFLEN];
  2
  2
  2          CNF$TRACE (DBG$C_TRACE, $DESCRIPTOR('TRACE'),
  2               $DESCRIPTOR ('cnf$bufr_err_msg'));
  2
  2          MSG = 0;                          ! Zero descriptor length and type fields
  2          ERR_TXTLEN = 0;
  2
  2          !
  2          !    Set up descriptor and buffer for decoding the optional error code
  2          !
  2          DECODED_TXT_LEN = 0;
  2          DECODED_TXT_BUFDSC = 0;
  2          DECODED_TXT_BUFDSC [DSC$W_LENGTH] = DECODED_TXT_BUFLEN;
  2          DECODED_TXT_BUFDSC [DSC$A_POINTER] = DECODED_TXT_BUF;
  2
  2          IF NOT NULLPARAMETER (4)
  2          THEN
  2              !
  2              !    Parameter ERR_TXT_COD was provided so decode it
  2              !
  3              BEGIN
P 3              $GETMSG (MSGID = .ERR_TXT_COD,
P 3                       MSGLEN = DECODED_TXT_LEN,
  3                       BUFADR = DECODED_TXT_BUFDSC);
  3              ERR_TXTLEN = .DECODED_TXT_LEN;
  3              END
  2          ELSE
  2              !
  2              !    Optional parameter ERR_TXT_COD was not provided so see if
  2              !    ERR_TXT_DSC was and use it instead.
  2              !
  3              BEGIN
  3              IF NOT NULLPARAMETER (5)
  3              THEN
  3                  ERR_TXTLEN = .ERR_TXT_DSC [DSC$W_LENGTH];
  2              END;
  2
  2          MSG [DSC$W_LENGTH] = 4 + .ERR_TXTLEN;
  2          EXECUTE (CNF$GET_ZVM (MSG [DSC$W_LENGTH], MSG [DSC$A_POINTER]) );    ! Get space to store message
  2
  2          (.MSG [DSC$A_POINTER]) <0, 8> = .ERR_CODE;           ! First byte is error code
  2          (.MSG [DSC$A_POINTER]) <8, 16> = .ERR_DETAIL;        ! Second and third bytes are error detail
  2          (.MSG [DSC$A_POINTER]) <24, 8> = .ERR_TXTLEN;        ! Fourth byte is length of optional error text
  2
  2          IF .ERR_TXTLEN GTR 0
  2          THEN
  2              !
  2              !    Ooptional text was provided either by decoding ERR_TXT_COD
  2              !    or in ERR_TXT_DSC, so append it to error message being built
  2              !
```

```
;   287          0470  2           CH$MOVE (.ERR_TXTLEN,
;   288          0471  3                   (IF .DECODED_TXT_LEN GTR 0
;   289          0472  3                    THEN
;   290          0473  3                         DECODED_TXT_BUF
;   291          0474  3                    ELSE
;   292          0475  3                         .ERR_TXT_DSC [DSC$A_POINTER]
;   293          0476  2                   ),
;   294          0477  2                   (.MSG [DSC$A_POINTER]) + 4);
;   295          0478  2
;   296          0479  2       CNF$BUFR_NICE_MSG (.IRB, MSG, .MSG [DSC$W_LENGTH]); ! Place the error message in the IRB for later trans
;   297          0480  2       RETURN TRUE;
;   298          0481  1       END;                                                ! Routine cnf$bufr_err_msg


                                                                      .PSECT  $PLIT$,NOWRT,NOEXE,2

                          45  43  41  52  54  0002C P.AAF:    .ASCII  \TRACE\                                             ;
                                              00031           .BLKB   3
                                   00000005   00034 P.AAE:    .LONG   5                                                   ;
                                   00000000'  00038           .ADDRESS P.AAF                                              ;
73  6D  5F  72  72  65  5F  72  66  75  62  24  66  6E  63  0003C P.AAH:    .ASCII  \cnf$bufr_err_msg\                    ;
                                              67  0004B                                                                   ;
                                   00000010   0004C P.AAG:    .LONG   16                                                  ;
                                   00000000'  00050           .ADDRESS P.AAH                                              ;

                                                              .EXTRN  SYS$GETMSG

                                                              .PSECT  $CODE$,NOWRT,2

                                        003C 00000            .ENTRY  CNF$BUFR_ERR_MSG, Save R2,R3,R4,R5              ; 0357
                    5E        FEEC   CE   9E 00002            MOVAB   -276(SP), SP
                              0000'  CF   9F 00007            PUSHAB  P.AAG                                           ; 0422
                              0000'  CF   9F 0000B            PUSHAB  P.AAE                                           ; 0421
                                     01   DD 0000F            PUSHL   #1                                              ;
               0000G  CF             03   FB 00011            CALLS   #3, CNF$TRACE                                   ;
                              F8     AD   D4 00016            CLRL    MSG                                             ; 0424
                                     52   D4 00019            CLRL    ERR_TXTLEN                                      ; 0425
                                     6E   D4 0001B            CLRL    DECODED_TXT_LEN                                 ; 0430
                              F0     AD   D4 0001D            CLRL    DECODED_TXT_BUFDSC                              ; 0431
              F0  AD        0100     8F   B0 00020            MOVW    #256, DECODED_TXT_BUFDSC                        ; 0432
              F4  AD          04     AE   9E 00026            MOVAB   DECODED_TXT_BUF, DECODED_TXT_BUFDSC+4           ; 0433
                  04                 6C   91 0002B            CMPB    (AP), #4                                        ; 0435
                                     1D   1F 0002E            BLSSU   1$                                              ;
                              10     AC   D5 00030            TSTL    16(AP)                                          ;
                                     18   13 00033            BEQL    1$                                              ;
                  7E                 0F   7D 00035            MOVQ    #15, -(SP)                                      ; 0443
                              F0     AD   9F 00038            PUSHAB  DECODED_TXT_BUFDSC                              ;
                              0C     AE   9F 0003B            PUSHAB  DECODED_TXT_LEN                                 ;
                              10     AC   DD 0003E            PUSHL   ERR_TXT_COD                                     ;
          00000000G  00              05   FB 00041            CALLS   #5, SYS$GETMSG                                  ;
                  52                 6E   D0 00048            MOVL    DECODED_TXT_LEN, ERR_TXTLEN                     ; 0444
                                     0E   11 0004B            BRB     2$                                              ; 0435
                  05                 6C   91 0004D 1$:        CMPB    (AP), #5                                        ; 0452
                                     09   1F 00050            BLSSU   2$                                              ;
                              14     AC   D5 00052            TSTL    20(AP)                                          ;
                                     04   13 00055            BEQL    2$                                              ;
```

```
                           52          14    BC  3C 00057          MOVZWL  @ERR_TXT_DSC, ERR_TXTLEN                          ; 0454
          F8    AD         52                04  A1 0005B 2$:      ADDW3   #4, ERR_TXTLEN, MSG                               ; 0457
                                       FC    AD  9F 00060          PUSHAB  MSG+4                                             ; 0458
                                       F8    AD  9F 00063          PUSHAB  MSG                                               ;
                   0000G   CF                02  FB 00066          CALLS   #2, CNF$GET_ZVM                                   ;
                           3E                50  E9 0006B          BLBC    STATUS, 6$                                        ;
                           51          FC    AD  D0 0006E          MOVL    MSG+4, R1                                         ; 0460
                           61          08    AC  90 00072          MOVB    ERR_CODE, (R1)                                    ;
                     01    A1          0C    AC  B0 00076          MOVW    ERR_DETAIL, 1(R1)                                 ; 0461
                     03    A1                52  90 0007B          MOVB    ERR_TXTLEN, 3(R1)                                 ; 0462
                                             52  D5 0007F          TSTL    ERR_TXTLEN                                        ; 0464
                                             17  15 00081          BLEQ    5$                                                ;
                                             6E  D5 00083          TSTL    DECODED_TXT_LEN                                   ; 0471
                                             06  15 00085          BLEQ    3$                                                ;
                           50          04    AE  9E 00087          MOVAB   DECODED_TXT_BUF, R0                               ;
                                             08  11 0008B          BRB     4$                                                ;
                           50          14    AC  D0 0008D 3$:      MOVL    ERR_TXT_DSC, R0                                   ; 0475
                           50          04    A0  D0 00091          MOVL    4(R0), R0                                         ;
          04    A1         60                52  28 00095 4$:      MOVC3   ERR_TXTLEN, (R0), 4(R1)                           ; 0477
                           7E          F8    AD  3C 0009A 5$:      MOVZWL  MSG, -(SP)                                        ; 0479
                                       F8    AD  9F 0009E          PUSHAB  MSG                                               ;
                                       04    AC  DD 000A1          PUSHL   IRB                                               ;
                   FF0F    CF                03  FB 000A4          CALLS   #3, CNF$BUFR_NICE_MSG                             ;
                           50                01  D0 000A9          MOVL    #1, R0                                            ; 0480
                                                 04 000AC 6$:      RET                                                       ; 0481

; Routine Size:  173 bytes,    Routine Base:  $CODE$ + 0048
```

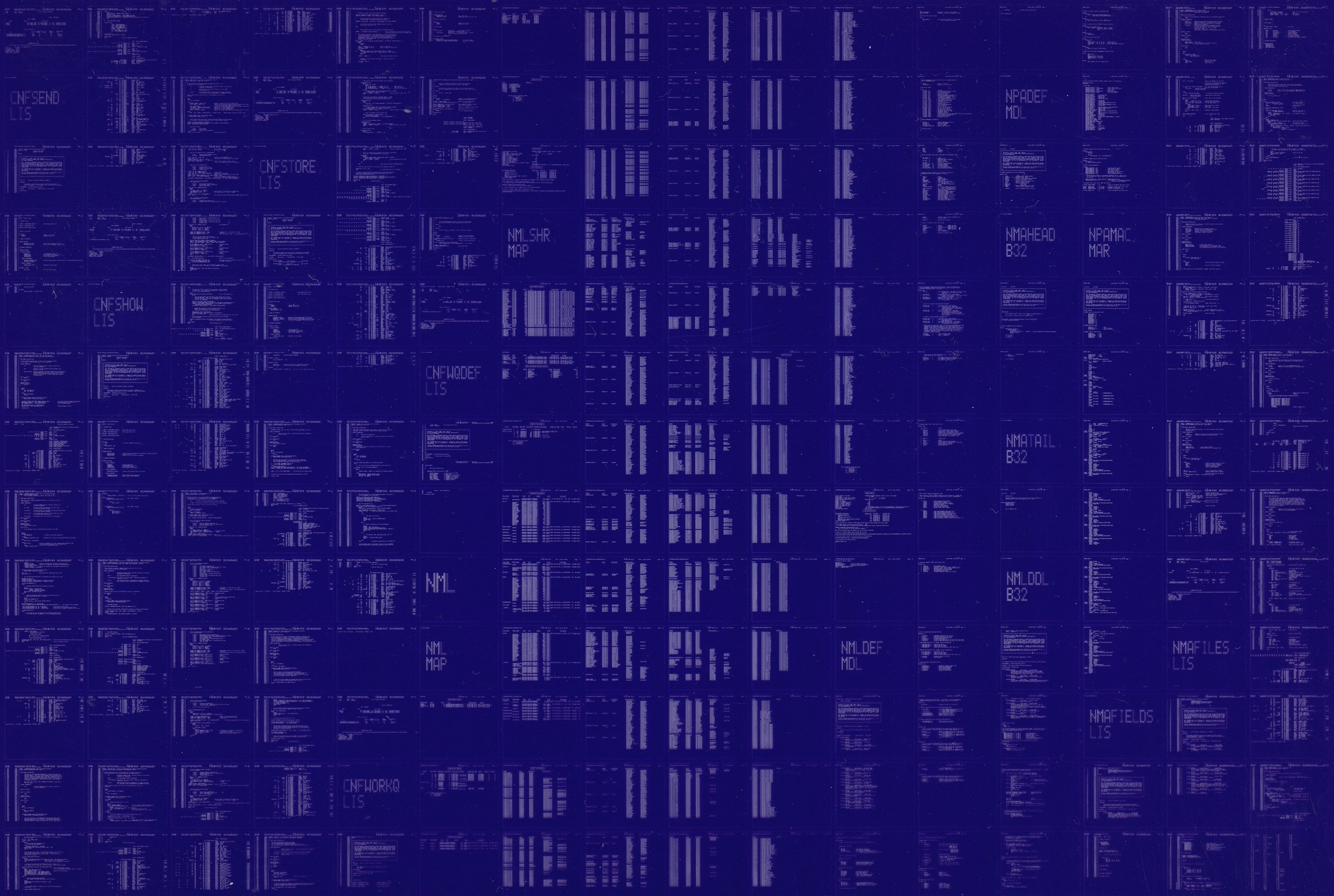
```
1 %SBTTL  'CNF$SEND_NICE_MSG  send the response message'
1 GLOBAL ROUTINE CNF$SEND_NICE_MSG (IRB) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Called first from CNF$PROCESS_REQUEST, a routine executed off
1 !       the work queue.  There will be an assumption at this point that
1 !       the IOSB contains a success from a previous interaction over the
1 !       channel.  The first NICE message in the IRB is QIO'd and from
1 !       then on CNF$SEND_NICE_MSG is executed as an AST routine upon QIO
1 !       completion.  The IOSB is checked before another NICE message
1 !       is removed and QIO'd.
1 !
1 !       When the list is empty then a CNF$SOLICIT_REQUEST is placed on
1 !       the work queue.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       irb                 Interrupt Request Block, contains context for
1 !                           I/O with connectee.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2    BEGIN
2    MAP
2       IRB : REF BBLOCK;
2
2    LOCAL
2       BNR : REF BBLOCK,
2       STATUS;
2
2
2    CNF$TRACE (DBG$C_TRACE, $DESCRIPTOR('TRACE'),
2       $DESCRIPTOR ('cnf$send_nice_msg'));
2
2       !
2       !   The first time thru, the IOSB should contain a success status
2       !   from a previous I/O on the channel.  For subsequent passes,
2       !   CNF$SEND_NICE_MSG will be called to send the next NICE message.
2       !   Then the IOSB will contain the status for the previous send,
2      !    and then if there was an error on the channel, the channel will
2      !    will be closed.
2      !
2      STATUS = .IRB [IRB$W_IOSB];
2      IF NOT .STATUS
2      THEN
3          BEGIN
3          IF (.STATUS NEQ SS$_LINKABORT) AND
4             (.STATUS NEQ SS$_LINKEXIT)
3          THEN
3              SIGNAL (CNF$_LINK, 0, .STATUS);
3          WKQ$ADD_WORK_ITEM (CNF$CLOSE_REQUEST_LINK, .IRB);
3          RETURN TRUE;
2          END;
2
2      !
2      !    Check to see if this call of the routine follows a call in which a
2      !    buffered message was sent.  In that case it should now be
2      !    deallocated.  This would not be the case if this was the first
2      !    call to this routine.
2      !
2      IF .IRB [IRB$W_FREE_LEN] NEQ 0
2      THEN
3          BEGIN
3          EXECUTE (CNF$FREE_VM (%REF(.IRB [IRB$W_FREE_LEN]), IRB [IRB$L_NICE_ADR]) );
3          IRB [IRB$W_FREE_LEN] = 0;          ! Keep it clean to avoid confusion
2          END;                               !  when another set of messages are buffered.
2
2      !
2      !    If there are any Buffered NICE Responses in the linked list
2      !    then remove the next and set it up for sending.  Deallocate the header.
2      !
2      IF .IRB [IRB$L_BNR_FLINK] NEQ IRB [IRB$L_BNR_FLINK]
2      THEN                                                        ! There are messages to send
3          BEGIN                                                   ! Get the next buffered message ready for sending
3          BNR = .IRB [IRB$L_BNR_FLINK];
3          REMQUE (.BNR, STATUS);                                  ! Remove the next message
3          IRB [IRB$W_NICE_LEN] = .BNR [BNR$W_LENGTH];
3          IRB [IRB$L_NICE_ADR] = .BNR [BNR$L_ADDRESS];
3          IRB [IRB$W_FREE_LEN] = .BNR [BNR$W_FREE_LEN];
3          EXECUTE (CNF$FREE_VM (%REF(BNR$C_LENGTH), BNR) );
3          END
2      ELSE
2          !
2          !    No more NICE messages buffered
2          !    Last request has been completed, solicit another.
2          !
3          BEGIN
3          WKQ$ADD_WORK_ITEM (CNF$SOLICIT_REQUEST, .IRB);
3          RETURN TRUE;
2          END;
2
2      !
2      !    If NICE debug logging is enabled, print the NICE message about
2      !    to be sent.
2      !
2      IF .CNF$GL_LOGMASK [DBG$C_NICE]
```

```
;  414          0596  2     THEN
;  415          0597  3         BEGIN
;  416          0598  3         LOCAL DATA_DSC : BBLOCK [DSC$C_S_BLN];
;  417          0599  3         DATA_DSC = 0;
;  418          0600  3         DATA_DSC [DSC$W_LENGTH] = .IRB [IRB$W_NICE_LEN];
;  419          0601  3         DATA_DSC [DSC$A_POINTER] = .IRB [IRB$L_NICE_ADR];
;  420          0602  3         CNF$LOG_DATA (DBG$C_NICE, $DESCRIPTOR ('NICE transmitted'),0, DATA_DSC);
;  421          0603  2         END;
;  422          0604  2
;  423          0605  2     !
;  424          0606  2     !   Send the NICE message
;  425          0607  2     !
;  426        P 0608  2     STATUS = $QIO
;  427        P 0609  2                 (
;  428        P 0610  2                 FUNC = IO$_WRITEVBLK,
;  429        P 0611  2                 CHAN = .IRB [IRB$W_CHAN],
;  430        P 0612  2                 EFN = CNF$C_ASYNCH_EFN,
;  431        P 0613  2                 IOSB = IRB [IRB$W_IOSB],
;  432        P 0614  2                 ASTADR = CNF$SEND_NICE_MSG,
;  433        P 0615  2                 ASTPRM = .IRB,
;  434        P 0616  2                 P1 = .IRB [IRB$L_NICE_ADR],
;  435        P 0617  2                 P2 = .IRB [IRB$W_NICE_LEN]
;  436          0618  2                 );
;  437          0619  2
;  438          0620  2     IF NOT .STATUS
;  439          0621  2     THEN SIGNAL (CNF$_LINK, 0, .STATUS);
;  440          0622  2
;  441          0623  2     RETURN TRUE;
;  442          0624  1     END;                            ! Routine cnf$send_nice_msg



                                                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

                                 45  43  41  52  54   00054 P.AAJ:   .ASCII  \TRACE\                                  ;
                                                      00059          .BLKB   3
                                          00000005    0005C P.AAI:   .LONG   5                                        ;
                                          00000000'   00060          .ADDRESS P.AAJ                                   ;
6D  5F  65  63  69  6E  5F  64  6E  65  73  24  66  6E  63   00064 P.AAL:   .ASCII  \cnf$send_nice_msg\              ;
                                                     67  73   00073                                                   ;
                                                      00075          .BLKB   3
                                          00000011    00078 P.AAK:   .LONG   17                                       ;
                                          00000000'   0007C          .ADDRESS P.AAL                                   ;
65  74  74  69  6D  73  6E  61  72  74  20  45  43  49  4E   00080 P.AAN:   .ASCII  \NICE transmitted\               ;
                                                         64   0008F                                                   ;
                                          00000010    00090 P.AAM:   .LONG   16                                       ;
                                          00000000'   00094          .ADDRESS P.AAN                                   ;

                                                                     .EXTRN  SYS$QIO

                                                                     .PSECT  $CODE$,NOWRT,2

                                            003C      00000          .ENTRY  CNF$SEND_NICE_MSG, Save R2,R3,R4,R5      ; 0483
                          55 00000000G  00    9E      00002          MOVAB   LIB$SIGNAL, R5                           ;
                          54 00000000G  8F    D0      00009          MOVL    #CNF$_LINK, R4                           ;
                          5E            10    C2      00010          SUBL2   #16, SP                                  ;
                                 0000'  CF    9F      00013          PUSHAB  P.AAK                                    ; 0532
                                 0000'   CF  9F 00017          PUSHAB  P.AAI                                          ; 0531
                                         01  DD 0001B          PUSHL   #1                                             ;
                     0000G  CF           03  FB 0001D          CALLS   #3, CNF$TRACE                                  ;
                            52     04    AC  D0 00022          MOVL    IRB, R2                                        ; 0542
                            53     0C    A2  32 00026          CVTWL   12(R2), STATUS                                 ;
                            23           53  E8 0002A          BLBS    STATUS, 2$                                     ; 0543
                 000020E4   8F           53  D1 0002D          CMPL    STATUS, #8420                                  ; 0546
                                         12  13 00034          BEQL    1$                                             ;
                 000020F4   8F           53  D1 00036          CMPL    STATUS, #8436                                  ; 0547
                                         09  13 0003D          BEQL    1$                                             ;
                                         53  DD 0003F          PUSHL   STATUS                                         ; 0549
                                         7E  D4 00041          CLRL    -(SP)                                          ;
                                         54  DD 00043          PUSHL   R4                                             ;
                            65           03  FB 00045          CALLS   #3, LIB$SIGNAL                                 ;
                                         52  DD 00048 1$:      PUSHL   R2                                             ; 0550
                                 0000G   CF  9F 0004A          PUSHAB  CNF$CLOSE_REQUEST_LINK                         ;
                                         55  11 0004E          BRB     6$                                             ;
                                   1E    A2  B5 00050 2$:      TSTW    30(R2)                                         ; 0560
                                         17  13 00053          BEQL    4$                                             ;
                                   20    A2  9F 00055          PUSHAB  32(R2)                                         ; 0563
                       04   AE     1E    A2  32 00058          CVTWL   30(R2), 4(SP)                                  ;
                                   04    AE  9F 0005D          PUSHAB  4(SP)                                          ;
                     0000G  CF           02  FB 00060          CALLS   #2, CNF$FREE_VM                                ;
                            01           50  E8 00065          BLBS    STATUS, 3$                                     ;
                                             04 00068          RET                                                    ;
                                   1E    A2  B4 00069 3$:      CLRW    30(R2)                                         ; 0564
                            50     14    A2  9E 0006C 4$:      MOVAB   20(R2), R0                                     ; 0571
                            50     14    A2  D1 00070          CMPL    20(R2), R0                                     ;
                                         29  13 00074          BEQL    5$                                             ;
                       04   AE     14    A2  D0 00076          MOVL    20(R2), BNR                                    ; 0574
                            53     04    BE  0F 0007B          REMQUE  @BNR, STATUS                                   ; 0575
                            51     04    AC  D0 0007F          MOVL    IRB, R1                                        ; 0576
                            50     04    AE  D0 00083          MOVL    BNR, R0                                        ;
                       1C   A1     08    A0  7D 00087          MOVQ    8(R0), 28(R1)                                  ;
                                   04    AE  9F 0008C          PUSHAB  BNR                                            ; 0579
                       04   AE           10  D0 0008F          MOVL    #16, 4(SP)                                     ;
                                   04    AE  9F 00093          PUSHAB  4(SP)                                          ;
                     0000G  CF           02  FB 00096          CALLS   #2, CNF$FREE_VM                                ;
                            0E           50  E8 0009B          BLBS    STATUS, 7$                                     ;
                                             04 0009E          RET                                                    ;
                                         52  DD 0009F 5$:      PUSHL   R2                                             ; 0587
                                 0000G   CF  9F 000A1          PUSHAB  CNF$SOLICIT_REQUEST                            ;
                     0000G  CF           02  FB 000A5 6$:      CALLS   #2, WKQ$ADD_WORK_ITEM                          ;
                                         60  11 000AA          BRB     9$                                             ; 0588
                            21   0000G   CF  E9 000AC 7$:      BLBC    CNF$GL_LOGMASK, 8$                             ; 0595
                                   08    AE  D4 000B1          CLRL    DATA_DSC                                       ; 0599
                            50     04    AC  D0 000B4          MOVL    IRB, R0                                        ; 0600
                       08   AE     1C    A0  B0 000B8          MOVW    28(R0), DATA_DSC                               ;
                       0C   AE     20    A0  D0 000BD          MOVL    32(R0), DATA_DSC+4                             ; 0601
                                   08    AE  9F 000C2          PUSHAB  DATA_DSC                                       ; 0602
                                         7E  D4 000C5          CLRL    -(SP)                                          ;
                                 0000'   CF  9F 000C7          PUSHAB  P.AAM                                          ;
                                         7E  D4 000CB          CLRL    -(SP)                                          ;
                     0000G  CF           04  FB 000CD          CALLS   #4, CNF$LOG_DATA                               ;
                                         7E  7C 000D2 8$:      CLRQ    -(SP)                                          ; 0618
                                         7E  7C 000D4          CLRQ    -(SP)                                          ;
                            50     04    AC  D0 000D6          MOVL    IRB, R0                                        ;
```

```
                        7E         1C  A0  32 000DA            CVTWL    28(R0), -(SP)                                   ;
                                   20  A0  DD 000DE            PUSHL    32(R0)                                          ;
                                       50  DD 000E1            PUSHL    R0                                              ;
                                 FF19  CF  9F 000E3            PUSHAB   CNF$SEND_NICE_MSG                               ;
                                   0C  A0  9F 000E7            PUSHAB   12(R0)                                          ;
                                       30  DD 000EA            PUSHL    #48                                             ;
                        7E         0A  A0  32 000EC            CVTWL    10(R0), -(SP)                                   ;
                           00000000G   8F  DD 000F0            PUSHL    #CNF$C_ASYNCH_EFN                               ;
             00000000G  00             0C  FB 000F6            CALLS    #12, SYS$QIO                                    ;
                        53             50  D0 000FD            MOVL     R0, STATUS                                      ;
                        09             53  E8 00100            BLBS     STATUS, 9$                                      ; 0620
                                       53  DD 00103            PUSHL    STATUS                                          ; 0621
                                       7E  D4 00105            CLRL     -(SP)                                           ;
                                       54  DD 00107            PUSHL    R4                                              ;
                        65             03  FB 00109            CALLS    #3, LIB$SIGNAL                                  ;
                        50             01  D0 0010C 9$:        MOVL     #1, R0                                          ; 0623
                                       04     0010F            RET                                                      ; 0624

; Routine Size:  272 bytes,    Routine Base:  $CODE$ + 00F5
;  444          0625  1 END                           ! End of module CNFSEND
;  445          0626  0 ELUDOM



                                                          .EXTRN  LIB$SIGNAL

;                               PSECT SUMMARY
;
;       Name                     Bytes                        Attributes
;
;    $PLIT$                          152  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;    $CODE$                          517  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                          Library Statistics
;
;                                        -------- Symbols --------        Pages        Processing
;       File                             Total    Loaded   Percent       Mapped        Time
;
;    _$255$DUA28:[SYSLIB]STARLET.L32;1    9776       11         0          581          00:01.1



;                               COMMAND QUALIFIERS
;
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:CNFSEND/OBJ=OBJ$:CNFSEND MSRC$:CNFSEND/UPDATE=(ENH$:CNFSEND)

; Size:          517 code + 152 data bytes
; Run Time:         00:12.4
; Elapsed Time:     00:28.1
; Lines/CPU Min:     3031
; Lexemes/CPU-Min: 22300
; Memory Used:  120 pages
; Compilation Complete
```

0280 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



CNFSEND
LIS

NPADEF
MDL

CNFSTORE
LIS

NMLSHR
MAP

NMAHEAD
B32

NPAMAC
MAR

CNFSHOW
LIS

CNFWQDEF
LIS

NMATAIL
B32

NML

NMLDDL
B32

NML
MAP

NMLDEF
MDL

NMAFILES
LIS

NMAFIELDS
LIS

CNFWORKQ
LIS